Operadores en Lenguaje C

Laboratorio de Programación

ATAJOS DE CODIGO

```
int a = 2;
int b = 3;
a += b;
```

Ya hemos visto la forma básica de asignar variables usando =, pero C ofrece algunos trucos abreviados adicionales para la asignación a los que llamamos “*atajos*”.
C nos permite agregar algo de forma abreviada al valor inicial de una variable usando +=. Puede hacer esta misma abreviatura con -=, *=, /= y %= para resta, multiplicación, división y módulo, respectivamente.

EJERCICIO 1

```
3  int main() {
4      //tiempo caida del servidor en minutos
5      //perdidas en dolares
6   int tiempoServidorCaido = 5;
7   double mediaPerdidas = 500.95;
8
9   return 0;
10  }
```

Calcule el total de pérdidas con respecto al tiempo de caída del servidor considerando que la media de perdidas e x 1 minuto. Se quiere reducir esa pérdida a la mitad obtenida.¿De qué monto total estamos hablando? Utilice atajos de código para resolver. Formatee la salida con 2 decimales.

Todavía no hemos visto las declaraciones que responden a valores verdadero/falso (conocidos como booleanos), pero como estamos hablando de símbolos aritméticos, vamos a ver una introducción a los símbolos utilizados para estas comprobaciones.

```
int a = 3;
int b = 3;
if (a == b) {
  a++;
}
```

Verificar si dos valores son iguales **==**

no es igual **!=**,

mayor que **>**,

mayor que o igual **>=**,

menor que **<** ,

y menor o igual que **<=**

Observe que hay doble = para verificar la equivalencia en lugar del símbolo único utilizado para la asignación

EJERCICIO 2

Utilice el operador lógico correcto

```
int main() {
  int x = 1;
  int y = 27;

  if (x == y) {
    printf("Felicitaciones! Ah usado los comparadores
        de forma correcta");
  } else {
    printf("Intente de nuevo!");
  }
    return 0;
}
```

EJERCICIO 3

Ahora intente con el siguiente código

```
int main() {
  int x = 5;  int y = 42;
  int z = 10;  int q = 5;
  int h = 90;  int j = 90;
  if (x  y) {
      if (z  q) {
          if (h j) {
   printf("Felicitaciones!! Ah usado los comparadores
       de forma correcta");
              }
          }
  }
    return 0;
}
```

Orden de Operadores

C examina las declaraciones y aplica reglas estándar al orden en que se deben procesar las operaciones. Por ejemplo, hará la multiplicación antes de la suma.

Mirando la tabla a continuación podrá ver que las operaciones con prioridad 1 se realizarán primero. Luego se procesa la prioridad 2, 3, etc.

Para operadores del mismo nivel de prioridad, las operaciones ocurren de izquierda a derecha.

Priority	Symbol
1	++
1	--
1	()
2	!
2	(typecast)
3	*
3	/
3	%
4	+
4	-
5	<, <=
5	>, >=
6	==, !=
7	&&
8	||
9	all assignment operators

EJERCICIO 4

Ahora intente con el siguiente código tratando de obtener el output de la derecha

```c
int main() {
  int x;
  int y;

  x = 2 + 3 * 5;
  y = 20 / 4 + 6;

  printf("x es: %d\n", x);
  printf("y es: %d\n", y);
    return 0;
}
```

```
x es: 25
y es: 2
```

FELICITACIONES!!! HEMOS AVANZADO MUCHO!

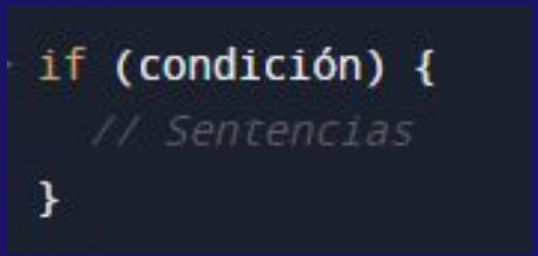
Hemos aprendido ya lo fundamental! te aliento a que dediques un tiempo a practicar!

Ya tenes conocimientos sobre: Operaciones matemáticas básicas en C: suma, resta, división, multiplicación, incremento, decremento y módulo. Asignación de valores a las variables: =, +=, -=, *=, /=, %=
Realizar comparaciones básicas entre valores y variables: ==, !=, <, <=, >, >=

continuemos!

If() { } Else { }

Una declaración if se usa para probar la verdad de una expresión y ejecutar algún código basado en ella. Aquí hay una forma simple de la declaración if:



Si la condición es verdadera, entonces se ejecutan las declaraciones dentro del bloque if. Cuando la condición es falsa, las declaraciones internas al fin se saltan y el programa continúa sin ejecutarlas.

If (){ } Else { }

La palabra clave if va seguida de un conjunto de paréntesis (). Dentro de estos paréntesis, se proporciona una condición que se evalúa como verdadera o falsa:

```
int main() {
    int flip = 1;
if (flip == 1) {
  printf("Aqui entro\n");
}
}
```

Si la condición se evalúa como verdadera, se ejecuta el código dentro de las llaves { }. Si la condición se evalúa como falsa, el código no se ejecuta.

Entonces, en el código anterior, si flip es igual a 1, el programa genera printf; si no es así, entonces no pasa nada y el programa continúa.

EJERCICIO 5

Podrías interpretar el siguiente código? Has comentarios en el código explicándolo.

```c
#include <stdlib.h>
#include <time.h>

int main() {
    srand (time(NULL));
   int coin = rand() % 2;

    if (coin == 0) {
     printf("\nCara\n");
   } else {
     printf("Cruz\n");
   }
}
```

Condicionales Múltiples

A veces necesitamos escribir múltiples condiciones en una declaración condicional utilizando los operadores lógicos.

- && **AND**
- || **OR**
- ! **NOT**

Cuando se colocan en una declaración condicional, los operadores lógicos trabajan juntos para producir una salida verdadera o falsa a través de múltiples condiciones.

Ejercicio 6!

El condicional anterior verifica si ambas condiciones son verdaderas, y si ambas lo son, el printf se imprime. Si *a* o *b* fuera un número negativo, el enunciado if sería falso.

```
int main() {
    int a = 1;
    int b = 2;

    if (a > 0 && b > 0) {
        printf("Se dió la condición\n");}

    return 0;
}
```

Cambia los valores de a y b para que sean iguales y modifica el condicional para que se imprima con ese cambio.